**WIL Report Management System**

Software design

­­­

By

**Ms. Phinthip Samutloiwon 552115050**

**Mr. Veerapat In-ongkarn 562115055**

Department of Software Engineering

College of Arts, Media and Technology

Chiang Mai University

Project Advisor

**Dr. Prompong Sugunnasil**

Document History

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| Document Name | Version | Status | | Date | Viewable | Reviewer | Responsible |
| Documents | | | | | | | |
| WRMS- SDD\_V.0.1.docx | * Add Chapter I   + Purpose   + Scope   + Acronym * Add Chapter II   + System architecture * Add Chapter III   + Class diagram   + Class diagram description   + Sequence diagram * Add Chapter IV   + UI design | | Draft | 18-02-2017 | PS,VI | PS,VI | PS,VI |
| WRMS- SDD\_V.0.2.docx | * Edit Chapter I   + Purpose   + Scope   + Acronym * Edit Chapter II * Edit Chapter III   + Class diagram   + Class diagram description   + Sequence diagram * Edit Chapter IV   + UI design | | Draft | 13-03-2017 | PS,VI | PS,VI, | PS,VI |
| WRMS- SDD\_V.0.3.docx | * Edit Chapter III   + Class diagram   + Class diagram description   + Sequence diagram | | Draft | 14-03-2017 | PS,VI | PS,VI, | PS,VI |

\*PS = Phinthip Samutloiwon  
\*VI = Veerapat In-ongkarn  
\*PSU= Prompong Sugunnasil

**Table of Contents**

[Chapter I | Introduction 4](#_Toc477207392)

[1. Purpose 4](#_Toc477207393)

[2. Scope 4](#_Toc477207394)

[3. Acronyms and Definition 4](#_Toc477207395)

[Chapter II | System Architecture 5](#_Toc477207396)

[Chapter III | Detailed design 6](#_Toc477207397)

[3.1 Class diagram 6](#_Toc477207398)

[3.2 Class diagram description 7](#_Toc477207399)

[CD-01: Users 7](#_Toc477207400)

[CD-02: Project 7](#_Toc477207401)

[CD-03: Task 8](#_Toc477207402)

[CD-04: TaskLog 8](#_Toc477207403)

[CD-05: UserService 9](#_Toc477207404)

[CD-06: ProjectService 10](#_Toc477207405)

[CD-07: TaskService 10](#_Toc477207406)

[CD-08: CommentService 10](#_Toc477207407)

[CD-09: NotificationService 10](#_Toc477207408)

[CD-10: UserController 11](#_Toc477207409)

[CD-11: AuthController 12](#_Toc477207410)

[CD-12: ProjectController 13](#_Toc477207411)

[CD-13: TaskController 14](#_Toc477207412)

[CD-14: CommentController 14](#_Toc477207413)

[CD-15: NotificationController 14](#_Toc477207414)

[3.3 Sequence diagram 15](#_Toc477207415)

[3.3.1 Feature 1 15](#_Toc477207416)

[3.3.2 Feature 2 19](#_Toc477207417)

[3.3.3 Feature 3 23](#_Toc477207418)

[3.3.4 Feature 4 23](#_Toc477207419)

[Chapter IV | UI Design 24](#_Toc477207420)

**Table of Figures**

[Figure 1 System architecture 6](#_Toc477268043)

[Figure 2 Class diagram of WRMS 7](#_Toc477268044)

[Figure 3 Sequence diagram of Registration 16](#_Toc477268045)

[Figure 4 Sequence diagram of Edit profile 17](#_Toc477268046)

[Figure 5 Sequence diagram of Login 18](#_Toc477268047)

[Figure 6 Sequence diagram of Logout 19](#_Toc477268048)

[Figure 7 Sequence diagram of Set code 19](#_Toc477268049)

[Figure 8 Sequence diagram of View project 20](#_Toc477268050)

[Figure 9 Sequence diagram of Add project 20](#_Toc477268051)

[Figure 10 Sequence diagram of Edit project 21](#_Toc477268052)

[Figure 11 Sequence diagram of Delete project 21](#_Toc477268053)

[Figure 12 Sequence diagram of View task 22](#_Toc477268054)

[Figure 13 Sequence diagram of Add task 22](#_Toc477268055)

[Figure 14 Sequence diagram of Edit task 23](#_Toc477268056)

[Figure 15 Sequence diagram of Delete task 23](#_Toc477268057)

[Figure 16 Sequence diagram of Move a task 24](#_Toc477268058)

[Figure 17 Login design 25](#_Toc477268059)

[Figure 18 Dashboard design 25](#_Toc477268060)

[Figure 19 Task overview list design 26](#_Toc477268061)

[Figure 20 Task overview card design 26](#_Toc477268062)

# **Chapter I | Introduction**

# **Purpose**

The purpose of this document is to plan the structure of the system follow from SRS. It will help developers to understand the overall system quicker and deeper detail. This document includes the software design of WIL Report management system which are System architecture, a system class diagram, detail description of class diagram, and sequence process of each use case.

# **Scope**

The scope of this document contain:

* Overall class diagram
* Detail of each class diagram including attribute and method
* Sequence diagram of each usecase
* UI design

# **Acronyms and Definition**

|  |  |
| --- | --- |
| Acronym | Stands for |
| SDD | Software design document |
| CD | Class diagram |
| SD | Sequence diagram |
| UI | User Interface |
| WRMS | WIL Report Management System |
| SRS | Software Requirement Specification |

# **Chapter II | System Architecture**

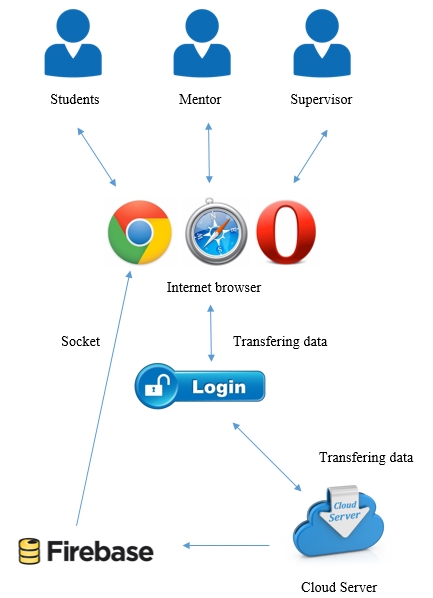


Figure 1 System architecture

# **Chapter III | Detailed design**

## **System class diagram**

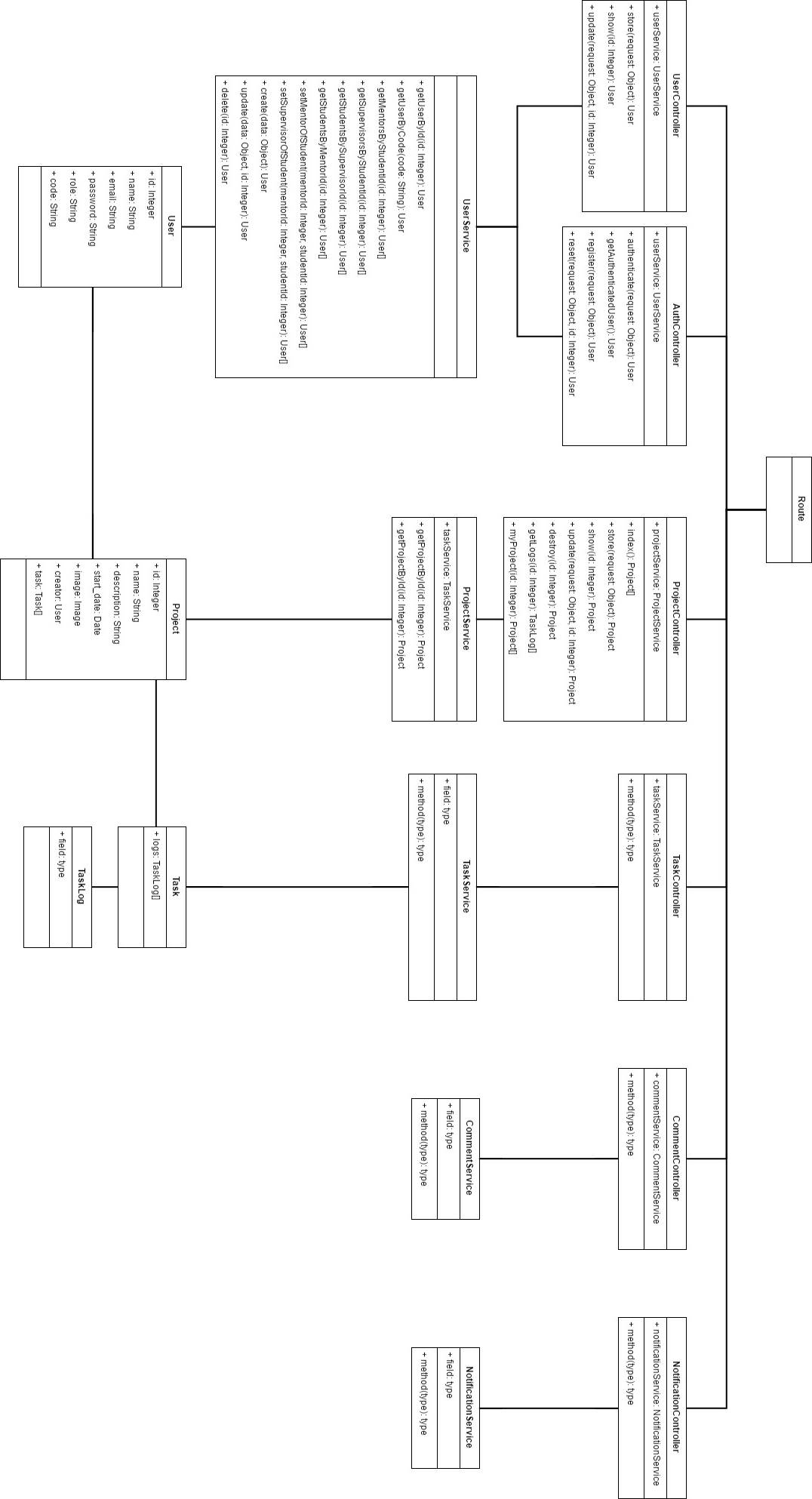
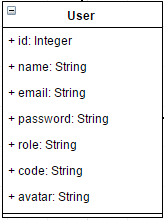


Figure 2 Class diagram of WRMS

## **Class diagram description**

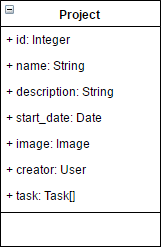
### **Users**



Entities

|  |  |  |  |
| --- | --- | --- | --- |
| ID | Name | Description | Type |
|  | id | Storing an id for identifying the user | Integer |
|  | name | Storing a name of the user | String |
|  | email | Storing an email of the user | String |
|  | password | Storing a password of the user | String |
|  | role | Storing the role of user | String |
|  | code | Storing the code of the user | String |
|  | avatar | Storing the path of profile image | String |

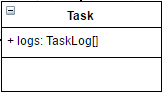
### **Project**



Entities

|  |  |  |  |
| --- | --- | --- | --- |
| ID | Name | Description | Type |
|  | id | Storing an id for identifying the project | Integer |
|  | name | Storing a name of the project | String |
|  | description | Storing description of the project | String |
|  | start\_date | Storing the start date of the project | Date |
|  | image | Storing an image of the project | image |
|  | creator | Storing an creator name of the project | User |
|  | task | Storing list of tasks | Task array |

### **Task**



Entities

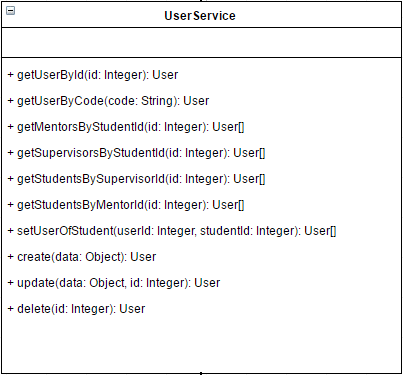
|  |  |  |  |
| --- | --- | --- | --- |
| ID | Name | Description | Type |
|  | logs | Storing the history of the task | TaskLog array |

### TaskLog

Entities

|  |  |  |  |
| --- | --- | --- | --- |
| ID | Name | Description | Type |
|  |  |  |  |

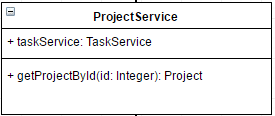
### **UserService**



Method Detail

|  |  |  |  |
| --- | --- | --- | --- |
| Method name | Description | Parameters | Return |
| getUserById(id: Integer): User | This method uses for getting the id of user. | id: integer | User Object |
| getUserByCode(code: String): User | This method uses for getting the user object by the code of user | code: string | User Object |
| getMentorsByStudentId(id: Integer): User[] | This method uses for getting the list of mentors by the student ID | id: integer | User Array |
| getSupervisorsByStudentId(id: Integer): User[] | This method uses for getting the list of supervisors by the student ID | id: integer | User Array |
| getStudentsBySupervisorId(id: Integer): User[] | This method uses for getting the list of students by the supervisor ID | id: integer | User Array |
| getStudentsByMentorId(id: Integer): User[] | This method uses for getting the list of students by Mentor ID | id: integer | User Array |
| setUserOfStudent(userId: Integer, studentId: Integer): User[] | This method uses for matching mentor and supervisor with student by inputting user ID(mentor and supervisor) and Student ID | Userid: integer  studentId: integer | User Array |
| create(data: Object): User | This method uses for creating the user object. | data: Object | User Object |
| update(data: Object, id: Integer): User | This method uses for updating the user object. | data: Object  id: Integer | User Object |
| delete(id: Integer): User | This method uses for deleting the user object. | id : integer | User Object |

### **ProjectService**



Entities

|  |  |  |  |
| --- | --- | --- | --- |
| ID | Name | Description | Type |
|  | taskService | Storing the task service in the project | TaskService Object |

Method Detail

|  |  |  |  |
| --- | --- | --- | --- |
| Method name | Description | Parameters | Return |
| getProjectById(id: Integer): Project | This method uses for getting the project object by id of the project. | id: integer | Project object |

### **TaskService**

Entities

|  |  |  |  |
| --- | --- | --- | --- |
| ID | Name | Description | Type |
|  |  |  |  |

Method Detail

|  |  |  |  |
| --- | --- | --- | --- |
| Method name | Description | Parameters | Return |
|  |  |  |  |

### **CommentService**

Entities

|  |  |  |  |
| --- | --- | --- | --- |
| ID | Name | Description | Type |
|  |  |  |  |

Method Detail

|  |  |  |  |
| --- | --- | --- | --- |
| Method name | Description | Parameters | Return |
|  |  |  |  |

### **NotificationService**

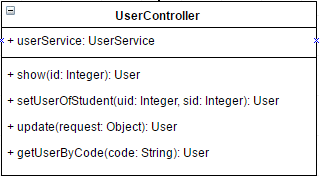
Entities

|  |  |  |  |
| --- | --- | --- | --- |
| ID | Name | Description | Type |
|  |  |  |  |

Method Detail

|  |  |  |  |
| --- | --- | --- | --- |
| Method name | Description | Parameters | Return |
|  |  |  |  |

### **UserController**



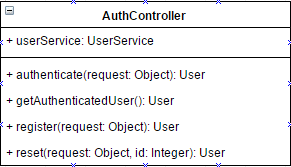
Entities

|  |  |  |  |
| --- | --- | --- | --- |
| ID | Name | Description | Type |
|  | userService | Object of userService, it uses to manage the user object by calling methods from userService class. | UserService Object |

Method Detail

|  |  |  |  |
| --- | --- | --- | --- |
| Method name | Description | Parameters | Return |
| show(id: Integer): User | This method uses for displaying the user object. | Id: the id of user | User Object |
| update(request: Object): User | This method uses for updating the new data of the user. | request: the object which contains the input data from user | User Object |
| getUserByCode(code: String): User | This method uses to get the user object by inputting the code of supervisor or mentor. | Code: the code of mentor or supervisor | User Object |
| setUserOfStudent(uid: Integer, sid: Integer): User | This method uses to set the mentor and supervisor of the student. | uid: the mentor and supervisor id  sid: the student id | User Object |

### **AuthController**



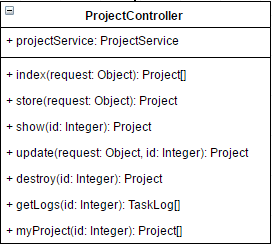
Entities

|  |  |  |  |
| --- | --- | --- | --- |
| ID | Name | Description | Type |
|  | userService | Object of userService, it uses to manage the user object by calling methods from userService class. | UserService Object |

Method Detail

|  |  |  |  |
| --- | --- | --- | --- |
| Method name | Description | Parameters | Return |
| authenticate(request: Object): User | This method uses for receiving username and password from ‘request object’, then find the use from database. | request | User Object |
| getAuthenticatedUser(): User | This method uses for getting the authenticated user object | - | User Object |
| register(request: Object): User | This method uses to receive the user information which called ‘request’, then create the user object in the database. | request | User Object |
| reset(request: Object, id: Integer): User | This method uses to reset the password. | request: Object | User Object |

### **ProjectController**



Entities

|  |  |  |  |
| --- | --- | --- | --- |
| ID | Name | Description | Type |
|  | projectService | Object of projectService, it uses to manage the project object by calling methods from projectService class. | ProjectService Object |

Method Detail

|  |  |  |  |
| --- | --- | --- | --- |
| Method name | Description | Parameters | Return |
| index(request: Object): Project[] | This method uses to get all project lists of the user. | request: the object which contain the owner data. | Project Array |
| store(request: Object): Project | This method uses to store the project data which called ‘request’ object in the database. | request: Object | Project Object |
| show(id: Integer): Project | This method uses to display the project object data which are project name, project description, start date, image, creator name, and tasks. | id: integer | Project Object |
| update(request: Object, id: Integer): Project | This method uses to update the project object data. | request: Object  id: integer | Project Object |
| destroy(id: Integer): Project | This method uses to remove the project object. | id: integer | Project Object |
| getLogs(id: Integer): TaskLog[] | This method uses to get the history of tasks. | id: integer | TaskLog Array |
| myProject(id: Integer): Project[] | This project uses to get the list of the projects of the owner. | id: integer | Project Array |

### **TaskController**

### **CommentController**

### **NotificationController**

## **Sequence diagram**

### **3.3.1 Feature 1**

1. **Register**

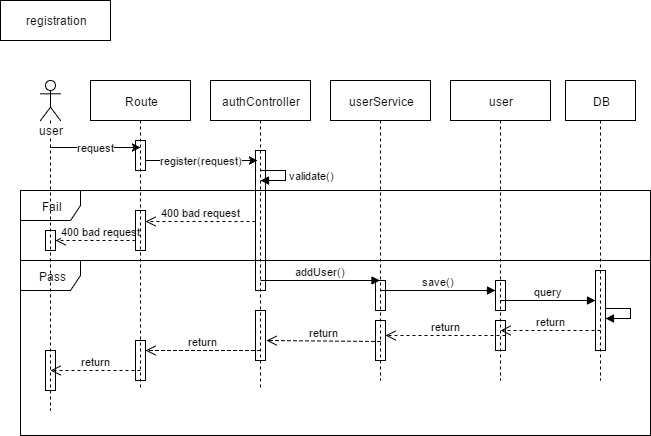


Figure 3 Sequence diagram of Registration

1. **Edit profile**

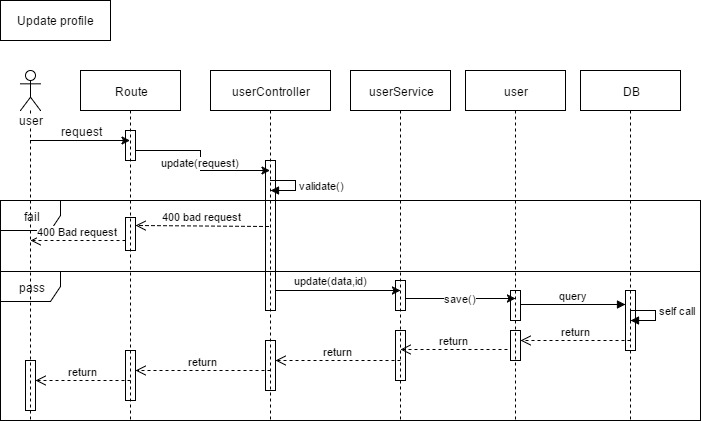


Figure 4 Sequence diagram of Edit profile

1. **Login**

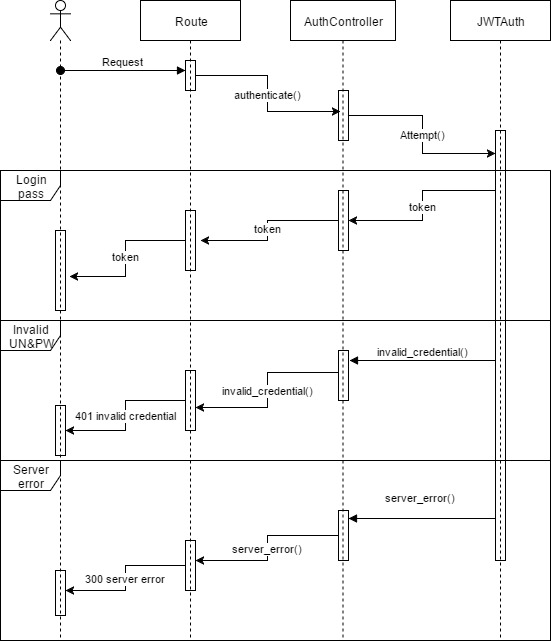


Figure 5 Sequence diagram of Login

1. **Logout**

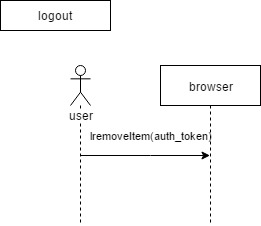


Figure 6 Sequence diagram of Logout

1. **Set code of supervisor or mentor**

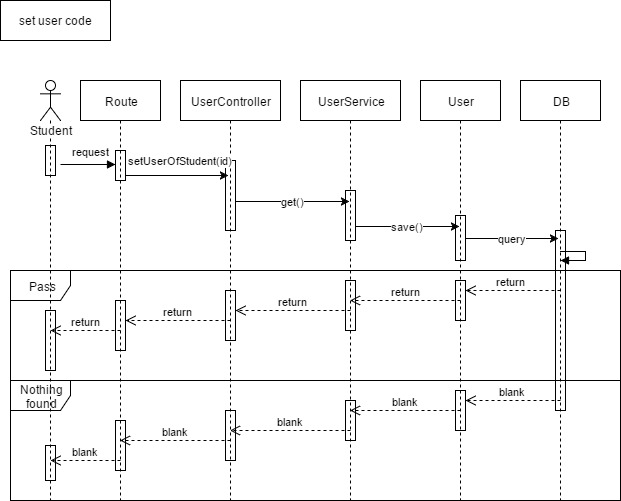


Figure 7 Sequence diagram of Set code

### **3.3.2 Feature 2**

1. **View project**

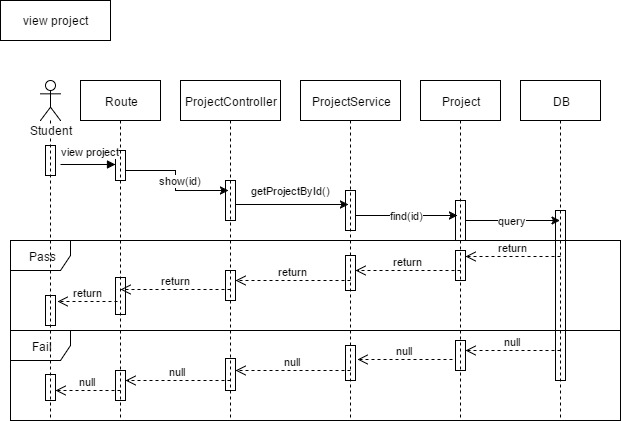


Figure 8 Sequence diagram of View project

1. **Add a project**

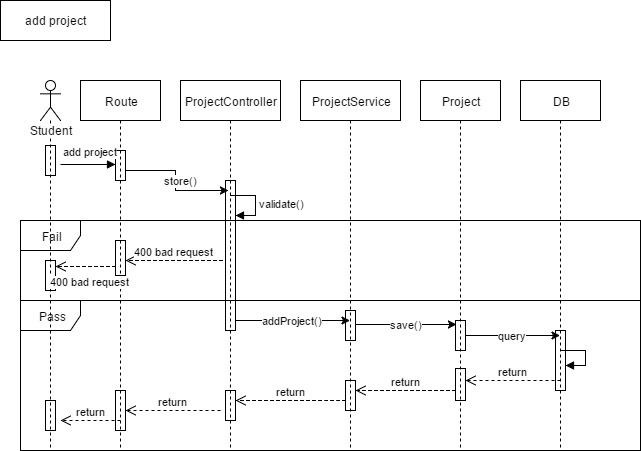


Figure 9 Sequence diagram of Add project

1. **Edit a project**

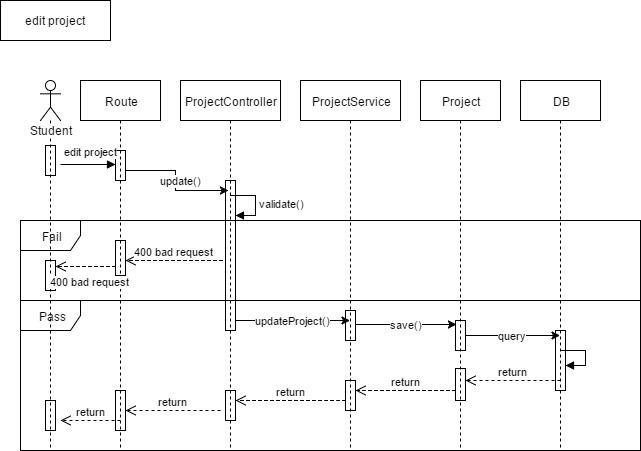


Figure 10 Sequence diagram of Edit project

1. **Delete a project**

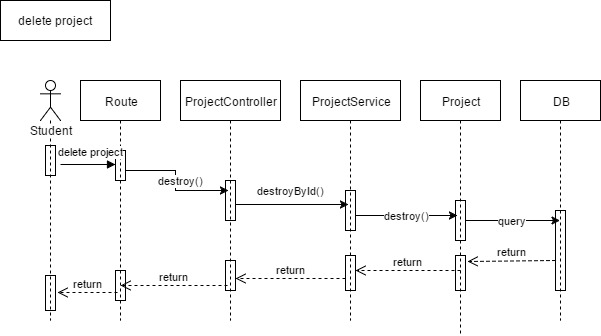


Figure 11 Sequence diagram of Delete project

1. **View a task**

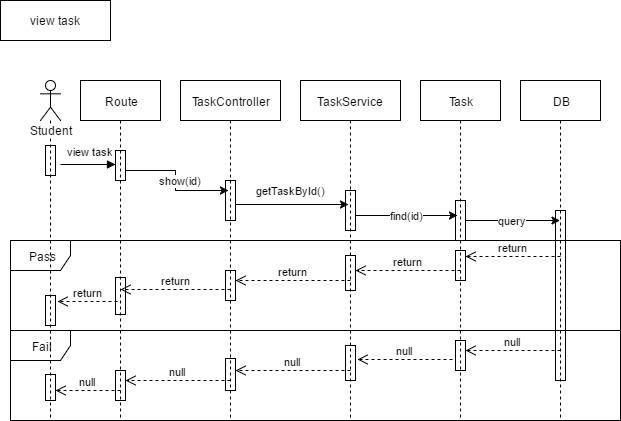


Figure 12 Sequence diagram of View task

1. **Add a task**

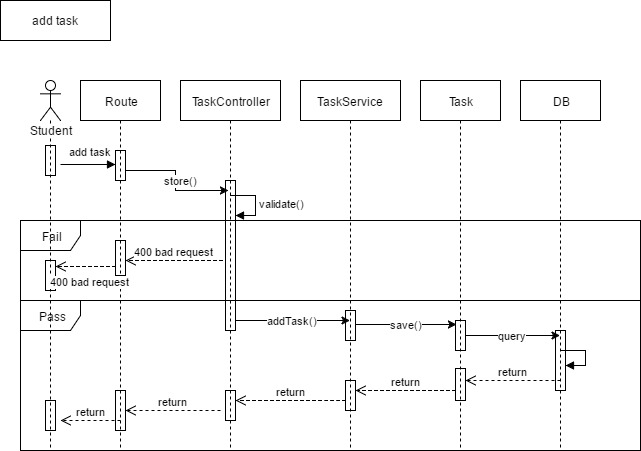


Figure 13 Sequence diagram of Add task

1. **Edit a task**

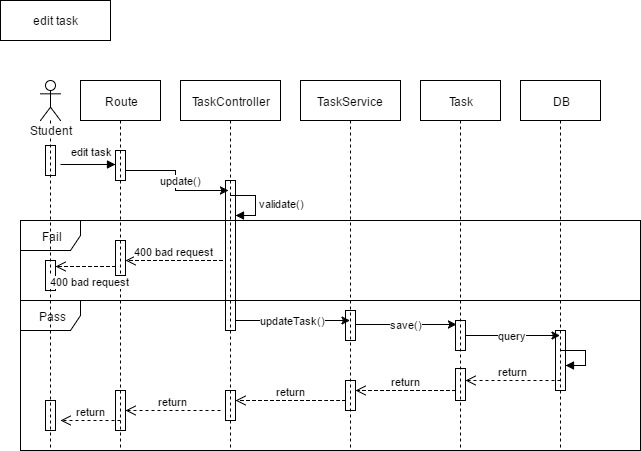


Figure 14 Sequence diagram of Edit task

1. **Delete a task**

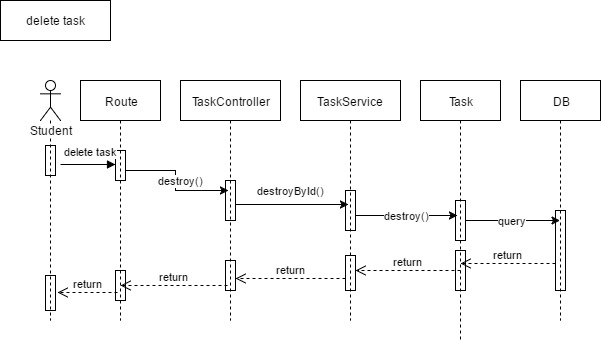


Figure 15 Sequence diagram of Delete task

1. **Move a task**

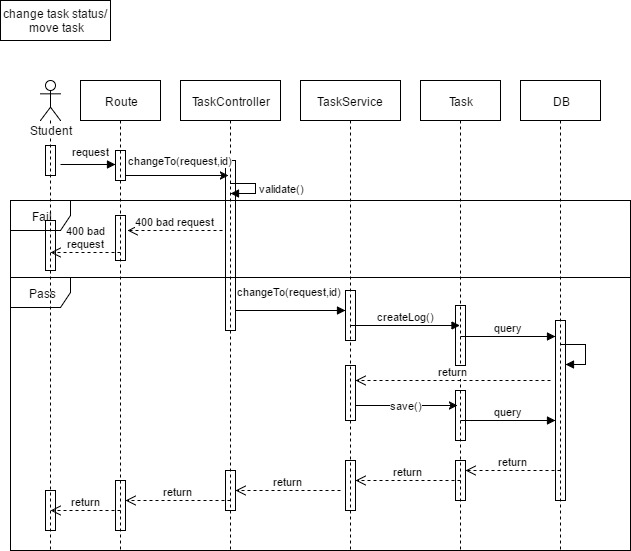


Figure Sequence diagram of Move a task

### 3.3.3 Feature 3

1. View a comment
2. Add a comment
3. Edit a comment
4. Delete a comment
5. View a weekly report
6. Generate a report

### 3.3.4 Feature 4

1. Receive a web notification
2. Receive an email notification

# **Chapter IV | UI Design**

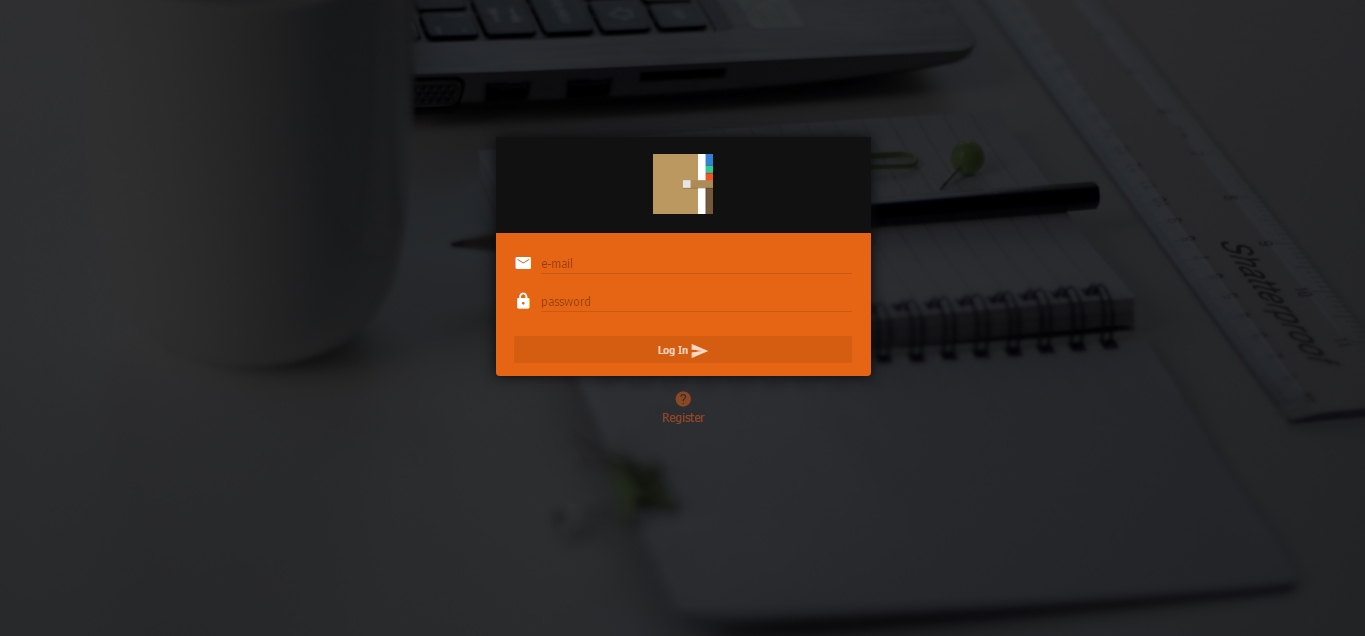


Figure 17 Login design

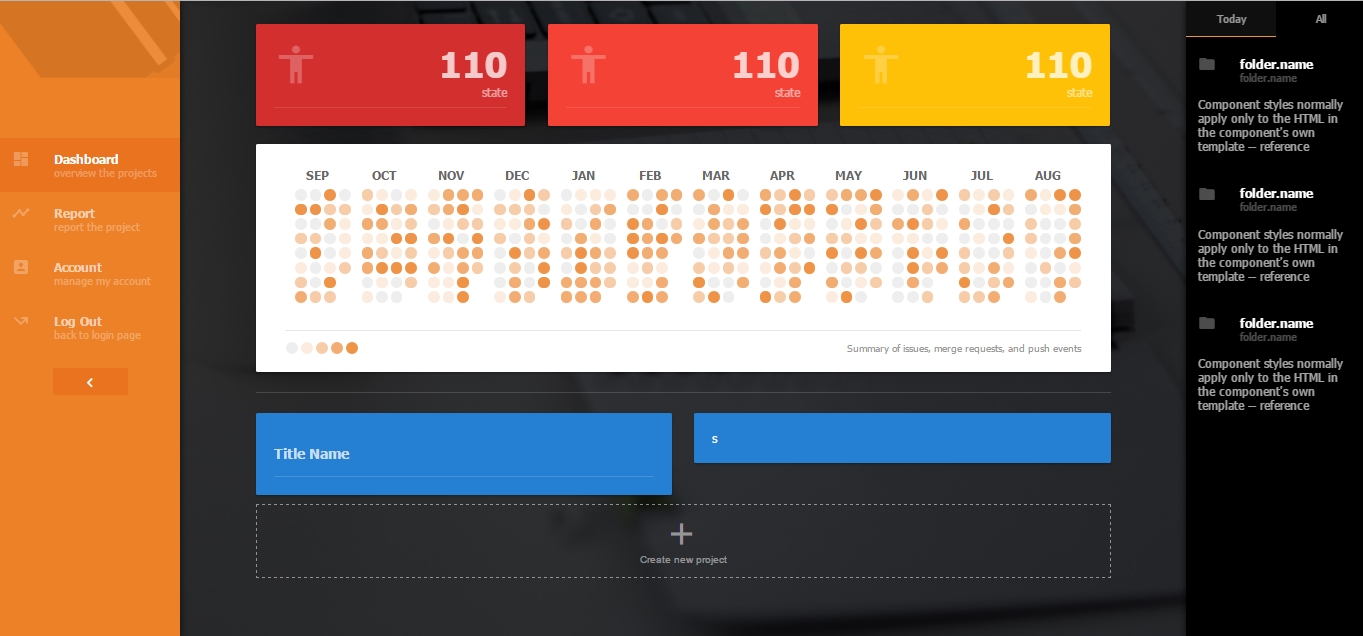


Figure 18 Dashboard design

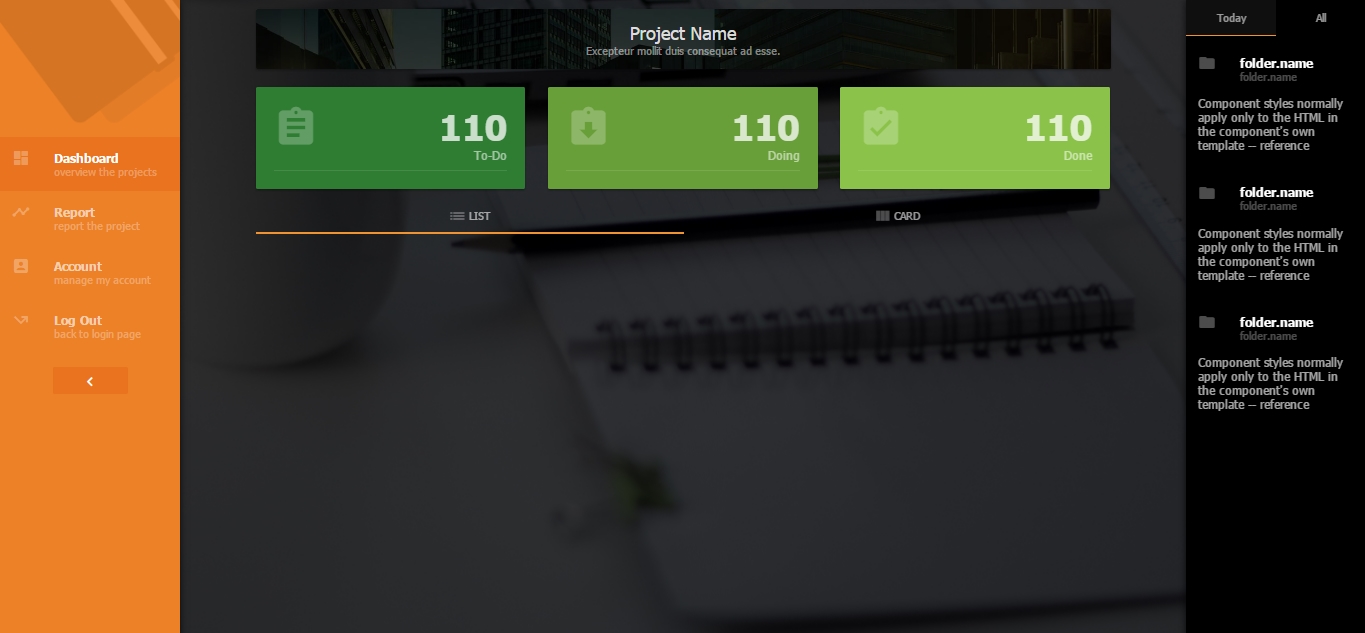


Figure 19 Task overview list design

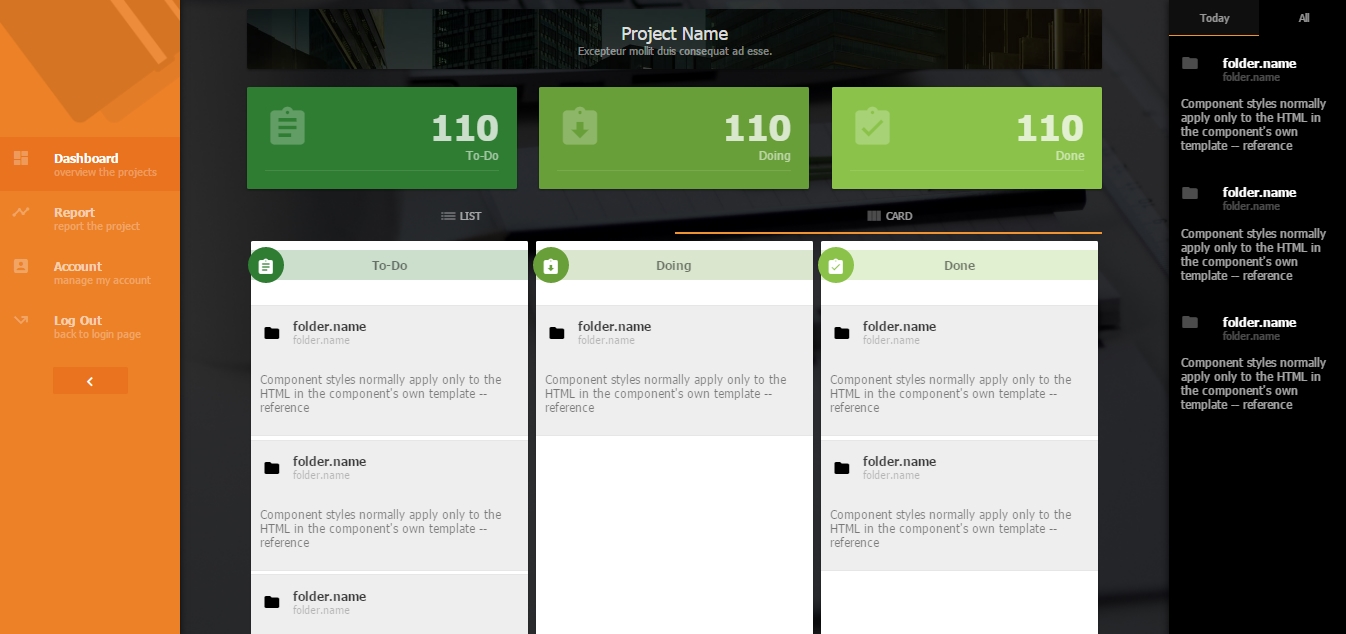


Figure 20 Task overview card design